**Ministerul Educației și Cercetării**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică și Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Raport**

Lucrarea de laborator nr.1

Disciplina: Programarea Aplicațiilor Distribuite

Tema: Agent de mesagerie

**Efectuat**: st.gr. TI-202 Bunescu Gabriel

**Efectuat**: st.gr. TI-202 Bodrug Mihai

**Efectuat**: st.gr. TI-202 Goliș Boris

**Verificat**: asist. univ. Cristofor Fistic

Chișinău 2023

Cuprins

[Introducere 3](#_Toc146926650)

[1 Concepte de bază 4](#_Toc146926651)

[1.1 Socket-uri 4](#_Toc146926652)

[1.1.1 Importanța Socket-urilor în comunicare 4](#_Toc146926653)

[1.2 gRPC 4](#_Toc146926654)

[1.2.1 Impactul în dezvoltarea de aplicații 5](#_Toc146926655)

[2 Implementarea agentului de mesaje cu socket 6](#_Toc146926656)

[2.1 Socket 6](#_Toc146926657)

[2.2 Implementarea agentului de mesaje cu gRPC 7](#_Toc146926658)

[3 Compararea soluțiilor 8](#_Toc146926659)

[Concluzii 10](#_Toc146926660)

[Bibliografie 11](#_Toc146926661)

# Introducere

În era digitală actuală, comunicarea în timp real este esențială pentru conectarea indivizilor și a dispozitivelor. Dezvoltarea unui agent de mesaje care facilitează trimiterea și primirea mesajelor în timp real reprezintă o provocare semnificativă. În acest raport, vom explora două abordări distincte pentru crearea unui astfel de agent de mesaje: una bazată pe socket-uri și cealaltă folosind tehnologia gRPC (Google Remote Procedure Call). Scopul acestui proiect este de a evidenția avantajele și dezavantajele fiecărei abordări, oferind o perspectivă cuprinzătoare asupra modurilor diferite de implementare a unei astfel de aplicații.

# Concepte de bază

În dezvoltarea unui agent de mesaje cu socket-uri și gRPC, este important să înțelegem conceptele de bază care stau la baza acestor două tehnologii.

## Socket-uri

În era digitală actuală, comunicarea între dispozitive și noduri este esențială pentru a facilita schimbul de informații într-o lume interconectată. Un concept fundamental în această comunicare este utilizarea socket-urilor. Socket-urile sunt o componentă-cheie în dezvoltarea aplicațiilor care permit transferul de date între două sau mai multe dispozitive prin intermediul rețelei.

La bază, un socket reprezintă un canal de comunicare bidirecțional între două dispozitive sau noduri, care pot fi situate în aceeași rețea locală sau la distanță. Socket-urile sunt utilizate pentru a permite dispozitivelor să trimită și să primească date în timp real. Acestea pot fi văzute ca puncte finale ale unei conexiuni de rețea și servesc ca interfață pentru aplicații în procesul de transmitere a datelor.

Există două tipuri principale de socket-uri:

* Socket-uri de server (Server Sockets) acestea sunt utilizate pentru a crea un punct final care ascultă conexiuni de la clienți. Un server socket așteaptă și acceptă cererile de conexiune de la clienți, permițându-le să se conecteze la server pentru a trimite și a primi date.
* Socket-uri de client (Client Sockets) acestea sunt utilizate de către clienți pentru a iniția o conexiune către un server. Clienții creează un socket client pentru a se conecta la un server socket și a iniția schimbul de date.

# Importanța Socket-urilor în comunicare

Socket-urile sunt esențiale pentru comunicarea în rețea din mai multe motive:

* comunicare bidirecțională a Socket-urile permit atât trimiterea cât și primirea de date între dispozitive, ceea ce le face ideale pentru interacțiuni în timp real;
* compatibilitate funcționează pe mai multe platforme și sisteme de operare, permițând dezvoltatorilor să creeze aplicații care pot comunica cu o varietate de dispozitive și servere;
* flexibilitate Socket-urile pot fi utilizate pentru a implementa o gamă largă de aplicații, de la mesagerie instantanee și jocuri multiplayer până la transfer de fișiere și streaming video;
* comunicarea prin socket-uri este eficientă și rapidă, deoarece oferă un canal de comunicare direct între dispozitive, fără a necesita intermediari suplimentari.

Socket-urile reprezintă o parte esențială a infrastructurii de comunicare în rețea, permițând dispozitivelor să interacționeze, să trimită și să primească date în timp real. Ele reprezintă un concept fundamental în dezvoltarea aplicațiilor moderne și au un rol crucial în conectarea lumii digitale de astăzi. Prin înțelegerea și utilizarea socket-urilor, dezvoltatorii pot crea aplicații puternice și scalabile care facilitează comunicarea și schimbul de informații între dispozitivele din întreaga lume.

### gRPC

În epoca noastră digitală în continuă evoluție, comunicarea eficientă între dispozitive și servicii este vitală. Un element cheie în revoluționarea acestui proces este gRPC, un framework dezvoltat cu măiestrie de Google, care deschide noi orizonturi în domeniul comunicării distribuite.

La bază, gRPC este un acronim pentru "Google Remote Procedure Call". Este un framework de comunicare care facilitează definirea și apelarea metodelor de la distanță între dispozitive sau servicii, folosind un protocol binar eficient. Acesta reprezintă un pas semnificativ în evoluția comunicării distribuite.

Caracteristici cheie ale gRPC:

* tipizarea mesajelor este unul dintre punctele forte ale gRPC. Acesta permite dezvoltatorilor să definească structura mesajelor folosind limbaje IDL (Interface Definition Language), cum ar fi Protocol Buffers, oferind astfel un nivel ridicat de claritate și consistență în comunicare;
* cu ajutorul limbajului IDL și a instrumentelor specifice, gRPC facilitează generarea automată a codului client-server. Aceasta reduce potențialele erori umane și accelerează procesul de dezvoltare;
* protocolul binar folosit de gRPC are o eficiență de transmitere a datelor mult mai mare în comparație cu alte formate, precum JSON sau XML. Acest lucru duce la o utilizare mai eficientă a lățimii de bandă și la performanță îmbunătățită;
* gRPC nu este legat de un limbaj de programare specific. Acesta oferă suport pentru o gamă variată de limbaje, cum ar fi Go, Python, Java, C++, și multe altele, ceea ce îl face versatil și ușor de integrat în proiectele existente;
* framework-ul oferă suport pentru autentificare și autorizare, permițând dezvoltatorilor să creeze aplicații securizate și de încredere.

# Impactul în dezvoltarea de aplicații

Utilizarea gRPC a transformat modul în care dezvoltatorii creează aplicații moderne. Această tehnologie aduce avantaje majore în dezvoltarea de microservicii, aplicații IoT, și în orice scenariu care necesită comunicare eficientă între diferite componente. Tipizarea mesajelor și generarea automată a codului conduc la dezvoltarea rapidă și la reducerea erorilor. Eficiența protocolului binar face gRPC o alegere excelentă pentru aplicații cu lățime de bandă limitată.

gRPC este un avans notabil în domeniul comunicării distribuite. Acest framework oferă dezvoltatorilor unelte puternice pentru a crea aplicații moderne, rapide și fiabile. Cu tipizare puternică, generare automată a codului, și suport pentru multiple limbaje de programare, gRPC rămâne la vârful tehnologiilor de comunicare în era digitală actuală. Utilizarea sa în dezvoltarea de aplicații continuează să contribuie la transformarea modului în care dispozitivele și serviciile interacționează, deschizând noi posibilități în lumea tehnologiei moderne.

# Implementarea agentului de mesaje cu socket

Pentru a dezvolta un agent de mesaje cu socket-uri, utilizăm limbajul de programare Python, această aplicație are drept scop permiterea comunicării în timp real între utilizatori și poate fi divizată în trei componente principale: serverul socket, clienții și modulul de gestionare a mesajelor.

## Socket

Serverul Socket:

* prima etapă constă în crearea unui server socket care ascultă pe o adresă IP și un port specific. Acest server va rămâne în așteptare pentru a accepta conexiuni de la clienți;
* clienții se conectează la server utilizând adresa IP și portul specificate. Serverul acceptă aceste conexiuni și creează un socket pentru fiecare client conectat;
* serverul primește mesaje de la clienți și se ocupă de distribuirea acestor mesaje către toți ceilalți clienți conectați. Astfel, orice mesaj trimis de un client este distribuit către toți ceilalți clienți, creând o comunicare bidirecțională în timp real.

Clienții:

* clienții sunt utilizatorii care se conectează la serverul de mesaje. Ei pot trimite și primi mesaje în timp real. Fiecare client are un socket dedicat pentru a comunica cu serverul.

Modul de gestionare a mesajelor este un modul separat sau o funcționalitate în server care poate gestiona mesajele primite. Acesta poate efectua operațiuni precum:

* decodarea și interpretarea mesajelor primite de la clienți;
* validarea și procesarea mesajelor înainte de distribuire;
* gestionarea mesajelor cu destinație specială sau restricționată;
* stocarea sau jurnalizarea mesajelor pentru referință ulterioară.

Implementarea unui agent de mesaje cu socket-uri necesită o abordare bine planificată și structurată. Este important să se asigure sincronizarea corectă între server și clienți pentru a evita probleme de conflict. De asemenea, gestionarea mesajelor și comunicarea în timp real pot adăuga complexitate la implementare și trebuie abordate cu atenție pentru a asigura o experiență de utilizare fluidă și sigură.

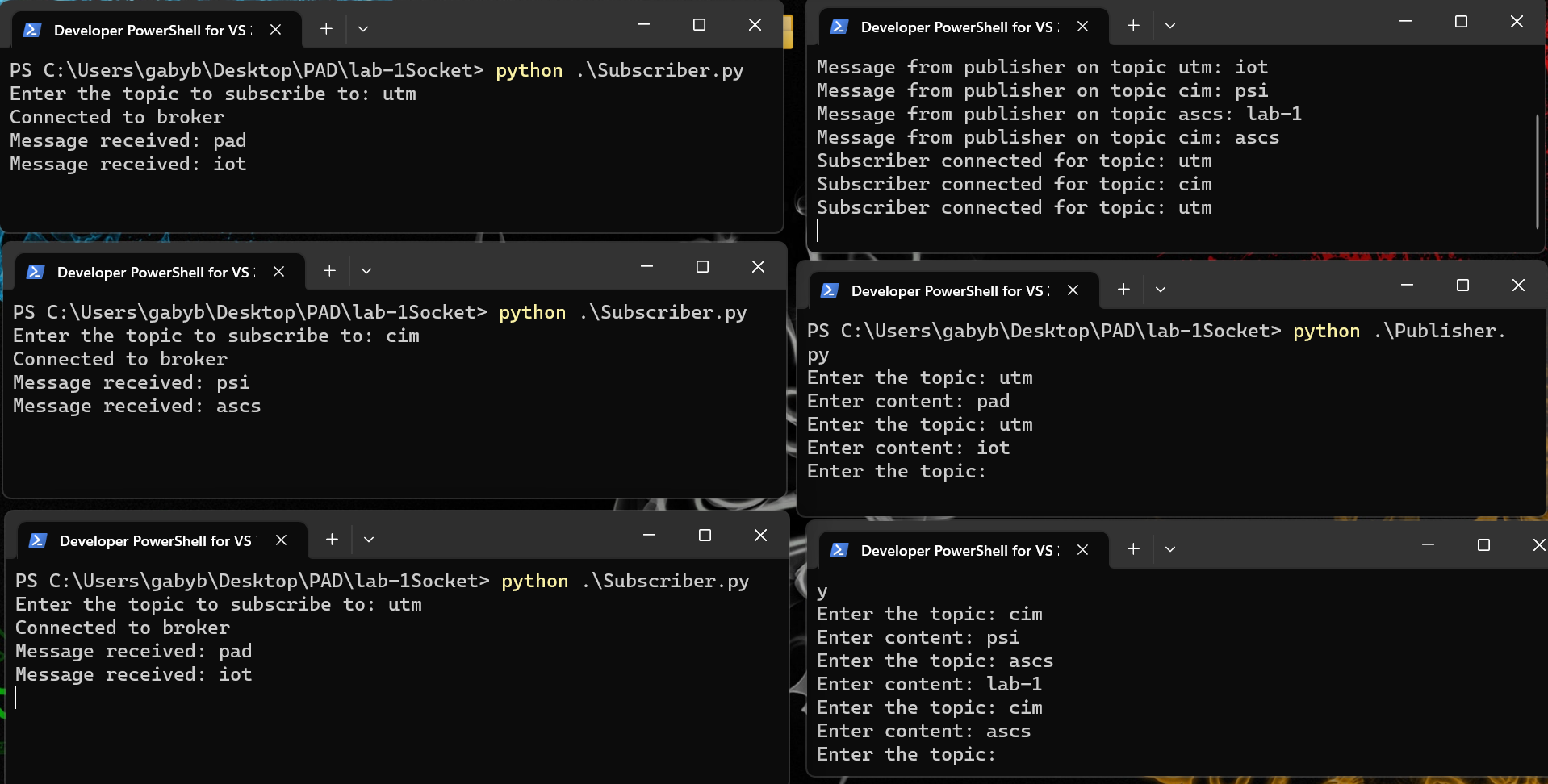


Figura 2.1 – Transmiterea și primirea a mesajul cu Socket.

## Implementarea agentului de mesaje cu gRPC

Pentru a dezvolta un agent de mesaje cu gRPC în limbajul de programare Python, am creat o aplicație care permite clienților să trimită și să primească mesaje în timp real. Acest agent se bazează pe trei componente principale: conexiuni, stocarea mesajelor și serviciile gRPC.

Conexiuni:

* pentru a gestiona conexiunile clienților, am definit structurile Connection și ConnectionStorage. Acestea permit adăugarea și eliminarea conexiunilor, precum și obținerea conexiunilor în funcție de topic. Fiecare conexiune are un canal gRPC dedicat pentru comunicare.

Stocarea mesajelor:

* pentru a gestiona mesajele, am creat clasa Message și MessageStorage. Acesta permite adăugarea de mesaje și obținerea următorului mesaj din coadă. Mesajele sunt stocate într-o listă pentru a fi distribuite ulterior către clienți.

Serviciile gRPC:

* am definit două servicii gRPC: PublisherService și SubscriberService. PublisherService permite clienților să publice mesaje către server, iar SubscriberService gestionează abonamentele clienților la topicuri specifice.

SenderWorker:

* componentă importantă este SenderWorker, care rulează într-un fir separat și se ocupă de distribuirea mesajelor către clienți. Acesta obține mesaje din stocarea mesajelor și le trimite către clienți pe canalele gRPC corespunzătoare.

Serverul gRPC:

* am creat un server gRPC care se ocupă de gestionarea serviciilor PublisherService și SubscriberService. Acest server ascultă pe portul 5001 și este configurat pentru a accepta conexiuni.

Clientul gRPC:

* un client gRPC simplu este inclus pentru a demonstra utilizarea serviciului PublisherService. Acest client permite utilizatorului să trimită mesaje la server.

ReceiverService:

* Pentru clienții care doresc să primească notificări, am creat un serviciu gRPC numit ReceiverService. Acest serviciu primește notificări și le afișează în consolă.

Aplicația principală:

* aplicația principală inițiază toate componentele și firurile necesare pentru funcționarea agentului de mesaje cu gRPC. Aceasta include gestionarea conexiunilor, stocarea mesajelor, serviciile gRPC și firul SenderWorker care distribuie mesajele.

Această implementare complexă a unui agent de mesaje cu gRPC oferă o perspectivă asupra modului în care se poate construi o aplicație de comunicare în timp real folosind această tehnologie. Agentul poate fi extins și personalizat în funcție de cerințele specifice ale proiectului pentru a asigura o comunicare eficientă și sigură între clienți și server.

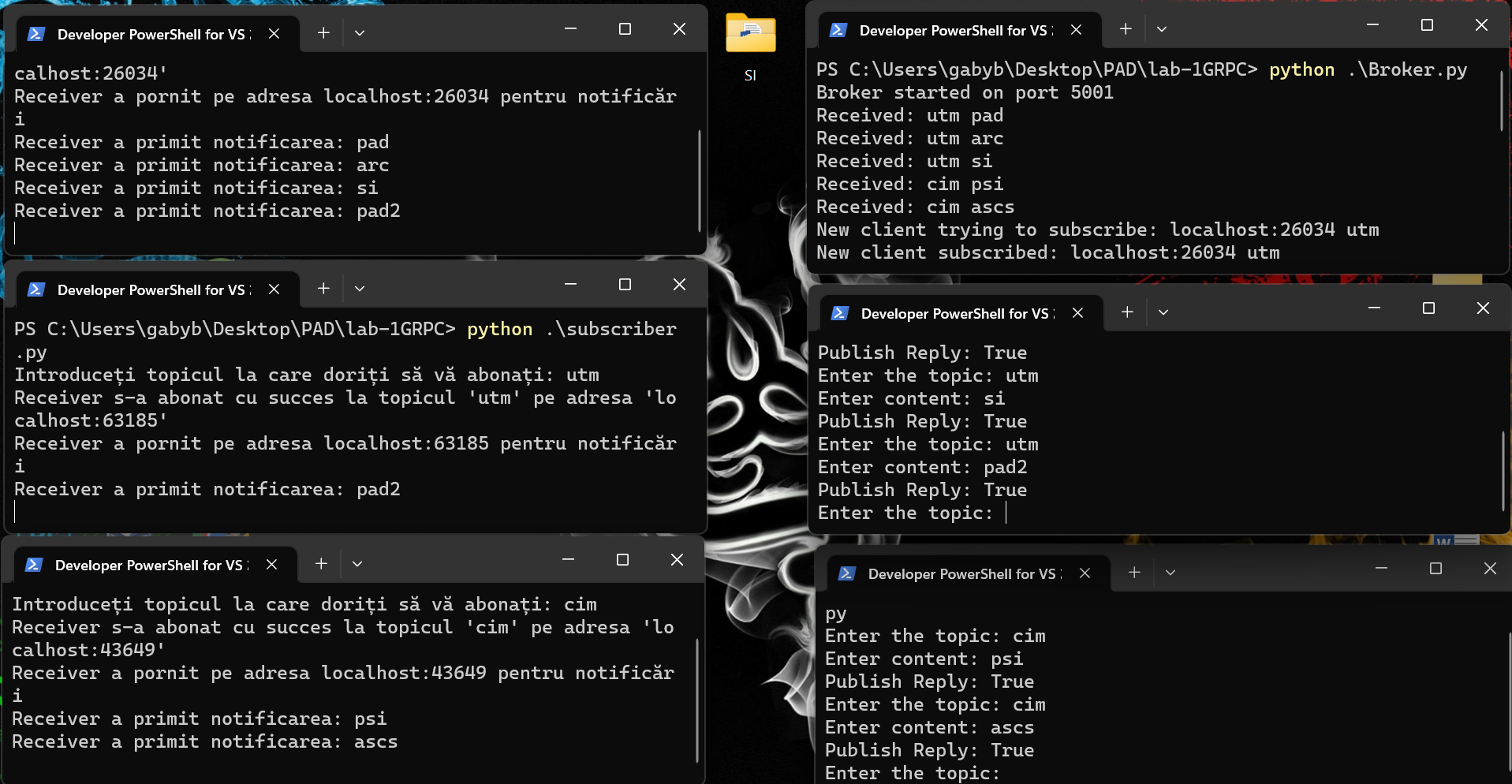


Figura 2.2 – Transmiterea și primirea a mesajul cu gRPC.

# Compararea soluțiilor

Implementarea unui sistem de comunicare în timp real poate fi abordată prin diverse tehnologii, vom analiza două soluții distincte: socket-uri și gRPC. Fiecare dintre acestea prezintă avantaje și dezavantaje care pot influența decizia de implementare.

Socket-urile sunt o modalitate tradițională și fundamentală de a realiza comunicarea între dispozitive într-o rețea.

gRPC este un framework dezvoltat de Google pentru comunicarea între sisteme distribuite. Această tehnologie aduce cu sine un set diferit de caracteristici.

| Criteriu | Avantaje Socket-uri | Dezavantaje Socket-uri | Avantaje gRPC | Dezavantaje gRPC |
| --- | --- | --- | --- | --- |
| Implementare | Simplă și ușor de înțeles | Gestionarea manuală a conexiunilor | Tipizare puternică a mesajelor | Complexitate inițială la definirea serviciilor și generarea codului |
| Tipizare Mesaje | Lipsa tipizării poate duce la erori | - | Evitarea erorilor comune datorită tipizării | - |
| Eficiență Transmitere | - | - | Eficiență ridicată în transmiterea datelor | - |
| Control Conexiuni | Gestionare manuală | - | Gestionare automată și eficientă a conexiunilor | - |
| Potrivit pentru LAN | Da | - | Da | - |
| Potrivit pentru WAN | Necesită gestionare mai complexă | - | Potrivit pentru WAN, poate fi mai eficient | - |
| Complexitate Inițială | - | - | - | Necesită învățarea noțiunilor specifice gRPC |
| Potrivit pentru Proiecte Mari | Mai puțin potrivit, datorită lipsei tipizării și a gestionării manuale | - | Potrivit pentru proiecte mari și complexe, datorită tipizării puternice și eficienței în transmitere | Complexitate inițială la definirea serviciilor și generarea codului |

Alegerea între Socket-uri și gRPC depinde de cerințele specifice ale proiectului. Socket-urile pot fi o opțiune bună pentru aplicații simple sau pentru scenarii în care controlul direct asupra conexiunilor este esențial. Pe de altă parte, gRPC oferă un nivel mai înalt de abstracție, cu tipizare puternică și eficiență în transmitere, ceea ce face ca această tehnologie să fie alegerea potrivită pentru aplicații complexe sau pentru comunicarea în sisteme distribuite. Este important să se ia în considerare nevoile specifice ale proiectului și experiența echipei de dezvoltare înainte de a lua o decizie finală.

# Concluzii

În cadrul acestui proiect, am realizat cu succes două variante ale unui agent de mesaje, folosind tehnologii distincte: socket-uri și gRPC. Această experiență ne-a permis să evaluăm în profunzime avantajele și dezavantajele fiecărei abordări, precum și criteriile de selecție între ele, în funcție de cerințele specifice ale proiectului.

Versiunea inițială a agentului de mesaje, bazată pe socket-uri, a reprezentat o abordare tradițională și familiară pentru comunicarea între dispozitive. Beneficiile acestei abordări includ:

* Socket-urile sunt ușor de înțeles și implementat, facilitând dezvoltarea rapidă;
* abordarea cu socket-uri este adecvată pentru comunicarea în rețele locale, unde controlul și performanța sunt critice.

Cu toate acestea, această abordare cu socket-uri vine și cu dezavantaje semnificative:

* dezvoltatorii trebuie să administreze manual deschiderea, închiderea și întreținerea conexiunilor, ceea ce poate deveni complex în aplicații cu mai mulți utilizatori;
* socket-urile nu oferă tipizare puternică a mesajelor, ceea ce poate conduce la erori dificil de identificat și corectat, mai ales în aplicații extinse și complexe.

Varianta a doua a agentului de mesaje, dezvoltată cu ajutorul gRPC, a introdus o abordare modernă pentru comunicarea în timp real. Beneficiile acestei abordări includ:

* gRPC oferă tipizare puternică a mesajelor, ceea ce reduce riscul de erori și sporește eficiența dezvoltării.
* protocolul binar și serializarea compactă fac transmiterea datelor eficientă și rapidă, un aspect crucial în aplicațiile cu volume mari de date.

Cu toate acestea, dezvoltarea cu gRPC poate presupune inițial o curba de învățare mai abruptă, necesitând familiarizarea cu conceptele specifice gRPC și definirea serviciilor.

Decizia între socket-uri și gRPC depinde în mare măsură de cerințele specifice ale proiectului. Dacă obiectivele includ implementarea rapidă sau comunicarea într-o rețea locală, socket-urile pot reprezenta o opțiune practicabilă. În schimb, pentru proiecte extinse și complexe, în care tipizarea puternică și eficiența transmiterii sunt esențiale, gRPC poate oferi avantaje semnificative.

Ca dezvoltatori, este esențial să evaluăm cu atenție cerințele proiectului și să alegem tehnologia potrivită în funcție de avantajele și dezavantajele fiecărei abordări. Această experiență ne-a îmbogățit cunoștințele și ne-a pregătit pentru dezvoltarea ulterioară a soluțiilor de comunicare în timp real.

# Bibliografie

<https://github.com/BunescuGabriel/PAD/tree/master/lab-1Socket>

<https://github.com/BunescuGabriel/PAD/tree/master/lab-1GRPC>

Начало формы